

```
UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP
UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP
UUU      UUU  EEEEEEEEEEEEEEE  TTTTTTTTTTTTTTT  PPPPPPPPPPPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEEEEEEEEEEEEEE  TTT      TTT
UUU      UUU  EEEEEEEEEEEEEEE  TTT      TTT
UUU      UUU  EEEEEEEEEEEEEEE  TTT      TTT
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUU      UUU  EEE      TTT      PPP      PPP
UUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT      TTT
UUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT      TTT
UUUUUUUUUUUUUUUU  EEEEEEEEEEEEEEE  TTT      TTT
```

```
RRRRRRRR  MM      MM  SSSSSSSS  TTTTTTTTTT  EEEEEEEEE  SSSSSSSS  TTTTTTTTTT  11
RRRRRRRR  MM      MM  SSSSSSSS  TTTTTTTTTT  EEEEEEEEE  SSSSSSSS  TTTTTTTTTT  11
RR      RR  MMMM  MMMM  SS      TT      EE      SS      TT      1111
RR      RR  MMMM  MMMM  SS      TT      EE      SS      TT      1111
RR      RR  MM    MM    SS      TT      EE      SS      TT      11
RR      RR  MM    MM    SS      TT      EE      SS      TT      11
RRRRRRRR  MM      MM  SSSSSS      EEEEEEE  SSSSSS      TT      11
RRRRRRRR  MM      MM  SSSSSS      EEEEEEE  SSSSSS      TT      11
RR      RR  MM      MM      SS      TT      EE      SS      TT      11
RR      RR  MM      MM      SS      TT      EE      SS      TT      11
RR      RR  MM      MM      SS      TT      EE      SS      TT      11
RR      RR  MM      MM      SSSSSSSS  TT      EEEEEEEEE  SSSSSSSS  TT      111111
RR      RR  MM      MM      SSSSSSSS  TT      EEEEEEEEE  SSSSSSSS  TT      111111

LL      IIIIII  SSSSSSSS
LL      IIIIII  SSSSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SSSSSS
LL      II     SSSSSS
LL      II     SS
LL      II     SS
LL      II     SS
LL      II     SS
LL      IIIIII  SSSSSSSS
LLLLLLLLL  IIIIII  SSSSSSSS
LLLLLLLLL  IIIIII  SSSSSSSS
```

```
0000 64          $BEGIN RMSTEST1,009,__,RMSTEST,<GENERAL RMS TEST PROGRAM>,<GBL, LONG>
0000 65
0000 66 ;
0000 67
0000 68          .ENABL  DBG
0000 69
0000 70
0000 71 :      this program tests the sequential file org, for a disk device,
0000 72 :      with vfc record format.
0000 73 :
0000 74 :
0000 75 :      test 1a:
0000 76 :
0000 77 :      create known test file of 1000 records, where each record has
0000 78 :      a 4-byte fixed control field giving the record # and a variable
0000 79 :      length portion of n mod 100 bytes of the ascii character n mod 10
0000 80 :      (where 'n' is the record number).
0000 81 :
0000 82
0000 83          $RMSDEF
0000 84          .NLIST  MEB
0000 85
0000 86 :
0000 87 :      macros:
0000 88 :
0000 89
0000 90          .MACRO  BUFF NAM,SIZE
0000 91 NAM'BUF::
0000 92          .BLKB   SIZE
0000 93          NAM'BSZ==SIZE
0000 94          .ENDM   BUFF
0000 95
0000 96 :
0000 97 :
0000 98 :
0000 99
0000 100         .MACRO  TYPE STRING, ?L
0000 101 STORE  <STRING>
0000 102 BLBC   VERBOSITY,L
0000 103 MOVL   $$$TMPX,CMDORAB+RAB$L RBF
0000 104 MOVW   $$$TMPX1,CMDORAB+RAB$W RSZ
0000 105 $PUT   RAB=CMDORAB,ERR=REPORT_ERROR
0000 106 BSBW   ERR
0000 107 L:
0000 108         .ENDM   TYPE
0000 109
0000 110 ;
0000 111
0000 112         .MACRO  WTTYPE STRING
0000 113 $WAIT   RAB=CMDORAB
0000 114 TYPE   <STRING>
0000 115         .ENDM   WTTYPE
0000 116
0000 117         .MACRO  WFIELD STRING
0000 118 $WAIT   RAB=CMDORAB
0000 119 FIELD  <STRING>
0000 120         .ENDM
```

RMSTEST1
V04-000

GENERAL RMS TEST PROGRAM ;

B 5

16-SEP-1984 01:45:37 VAX/VMS Macro V04-00
5-SEP-1984 04:21:39 [UETP.SRC]RMSTEST1.MAR;1

Page 2
(1)

0000 121
0000 122 ;
0000 123

RM
V0


```
0000 125 .MACRO STORE STRING,PRE
0000 126 .SAVE
0000 127 .PSECT $RMSNAM
0000 128 $$TMPX=-
0000 129 PRE ; store any carriage control info
0000 130 .ASCII %STRING%
0000 131 $$TMPX1=-$$TMPX
0000 132 .RESTORE
0000 133 .ENDM STORE
0000 134
0000 135 ;
0000 136
0000 137 .MACRO BEGIN TSTNAM
0000 138 STORE <TSTNAM>
0000 139 MOVL #$$TMPX,BEG_DESCR+4 ; addr
0000 140 MOVL #$$TMPX1,BEG_DESCR ; len
0000 141 BSBW BEGPUT
0000 142 .ENDM BEGIN
0000 143 .MACRO FINISH TSTNAM
0000 144 STORE <TSTNAM>
0000 145 MOVL #$$TMPX,FIN_DESCR+4 ; addr
0000 146 MOVL #$$TMPX1,FIN_DESCR ; len
0000 147 BSBW FINPUT
0000 148 .ENDM FINISH
0000 149 .MACRO FIELD FLDNAM
0000 150 STORE <FLDNAM>
0000 151 MOVL #$$TMPX,FLD_DESCR+4 ; addr
0000 152 MOVL #$$TMPX1,FLD_DESCR ; len
0000 153 BSBW FLDPUT
0000 154 .ENDM FIELD
0000 155 .MACRO MBPT, ?L
0000 156 BLBC VERBOSITY,L
0000 157 BPT
0000 158 L:
0000 159 .ENDM MBPT
0000 160
0000 161 ;
0000 162
```

```
00000000 164 .PSECT RMSTEST,GBL, LONG
0000 165 .ALIGN LONG
0000 166 T1START::
0000 167 WTRAB:: $RAB
00000364 0044 168 RFATBL: .BLKQ 100
0364 169 T1FAB:: $FAB FAC=PUT,FNM=<TST$DISK:T1FILE.DAT;1>,org=seq,rfm=vfc,-
0364 170 RAT=CR,FSZ=4,MRS=100,NAM=NAMBLK,FOP=<SUP,CTG>,-
0364 171 ALQ=48,DEQ=12,SHR=<PUT,GET,UPI>
03B4 172 T1RAB:: $RAB FAB=T1FAB,UBF=CPYBUF,USZ=CPYBSZ,RBF=CPYBUF,MBC=4,MBF=2,-
03B4 173 ROP=<WBH>,RHB=RECCNT,KBF=RECCNT
03F8 174 RECCNT::
00000000 00000000 03F8 175 .LONG 0,0
0400 176
0400 177 ;
0400 178
00000408'00000025' 0400 179 T1STR: .LONG T1L,T1S
44 52 4F 43 45 52 20 2E 4C 55 34 21 0408 180 T1S: .ASCII '!4UL. RECORDS WRITTEN. RFA = !XL,!XW'
20 20 2E 4E 45 54 54 49 52 57 20 53 0414
58 21 2C 4C 58 21 20 3D 20 41 46 52 0420
57 042C
00000025 042D 181 T1L=-T1S
042D 182
042D 183 ;
042D 184
00000435'0000005F' 042D 185 T2STR: .LONG T2L,T2S
20 3D 20 23 20 44 52 4F 43 45 52 20 0435 186 T2S: .ASCII 'RECORD # = !4UL, RHB = !4UL!/'
20 3D 20 42 48 52 20 2C 4C 55 34 21 0441
2F 21 4C 55 34 21 044D
4C 55 33 21 20 3D 20 45 5A 49 53 20 0453 187 .ASCII 'SIZE = !3UL, RSZ = !UW!/'
21 57 55 21 20 3D 20 5A 53 52 20 2C 045F
2F 046B
3D 20 41 46 52 20 44 45 56 41 53 20 046C 188 .ASCII 'SAVED RFA = !XL,!XW, FILE RFA = !XL,!XW'
49 46 20 2C 57 58 21 2C 4C 58 21 20 0478
4C 58 21 20 3D 20 41 46 52 20 45 4C 0484
57 58 21 2C 0490
0000005F 0494 189 T2L=-T2S
0494 190
0494 191 ;
0494 192
0000049C'0000000F' 0494 193 T3STR: .LONG T3L,T3S
46 41 21 27 3D 20 44 52 4F 43 45 52 049C 194 T3S: .ASCII 'RECORD = '!AF'!/'
2F 21 27 04A8
0000000F 04AB 195 T3L=-T3S
01 04AB 196 RHBSW: .BYTE 1 ; switch for modifying rhb contents
```

```

                                04AC 198 RMT$TEST_1A::
                                04AC 199 -WORD ^M<R2,R3,R4,R5,R6,R7,R8,R9,R10,R11>
                                04AE 200 BEGIN <SEQUENTIAL TESTS>
FB36 CF 00000002'EF BO 04C3 201 MOVW CMDORAB+RAB$W_ISI,WTRAB+RAB$W_ISI; copy output isi for
                                04CC 202
                                04CC 203 ;
                                04CC 204 ;DIFFERENT RAB $WAIT TEST
                                04CC 205 ;
                                04CC 206
FE93 CF 00100000 8F CA 04CC 207 BICL2 #FAB$M_CTG,T1FAB+FAB$L_FOP; don't want ctg set
      00000004'EF 01 CB 04D5 208 BISL2 #RAB$M_ASY,CMDORAB+RAB$L_ROP
      FF17 CF 01 DO 04DC 209 MOVL #1,RECCNT
      58 FB5F CF DE 04E1 210 MOVAL RFATBL,R8
      FB06' 30 04F7 211 $CREATE FAB=T1FAB,ERR=REPORT_ERROR
      5B FEB6 CF DE 04FA 212 BSBW ERR
      FAEF' 30 04FF 213 MOVAL T1RAB,R11
                                050E 214 $CONNECT RAB=R11,ERR=REPORT_ERROR
                                0511 215 BSBW ERR
                                0511 216
                                0511 217 ;
52 FEDE CF 00000064 8F 7B 0511 218
      56 051B 219 NXTREC: EDIV #100,RECCNT,R2,R6 ; compute record length
57 52 FED7 CF 0A 7B 051C 220 EDIV #10,RECCNT,R2,R7 ; compute character for record
      57 30 80 0523 221 ADDB #48,R7 ; make it ascii
      22 AB 56 B0 0526 222 MOVW R6,RAB$W_RSZ(R11) ; tell rms the record size
56 57 6E 00 2C 052A 223 MOVCS #0,(SP),R7,R6,CPYBUF ; fill record with character
      00000000'EF 052F 224 $PUT RAB=R11,ERR=REPORT_ERROR
      FABA' 30 0543 225 BSBW ERR
      30 57 D1 0546 226 CMPL R7,#48 ; is this 10th record?
      0B 12 0549 227 BNEQ T1CNT ; branch if not
      88 10 AB 7D 054B 228 MOVQ RAB$W_RFA(R11),(R8)+ ; save rfa in table
      56 05 054F 229 TSTL R6 ; is this 100th record?
      03 12 0551 230 BNEQ T1CNT
      093F 30 0553 231 BSBW TYPRFA ; type out the rfa
                                0556 232
                                0556 233 ;
                                0556 234
FFB1 FE9A CF 01 03E8 8F 3D 0556 235 T1CNT: ACBW #1000,#1,RECCNT,NXTREC
      FABC' 30 0560 236 $CLOSE FAB=T1FAB,ERR=REPORT_ERROR
      FE3E CF B4 0571 237 BSBW ERR
                                0574 238 CLRW T1RAB+RAB$W_ISI ; allow rab to be re-used
                                0578 239 $WAIT CMDORAB
                                0585 240 TYPE <FINISHED TEST 1A - CREATED TEST FILE>
```



```
05B4 242
05B4 243 :
05B4 244 : test 1b
05B4 245 :
05B4 246 : re-read file created in test 1a and try random and sequential
05B4 247 : access via $get and $find.
05B4 248 :
05B4 249 :
05B4 250 $WAIT RAB=CMDORAB
05C1 251 TYPE <BEGINNING TEST 1B - $GET AND $FIND>
FD6F CF 01000000 8F C8 05F0 252 BISL2 #FAB$M_NAM,T1FAB+FAB$FOP
FD7C CF 02 88 05F9 253 BISB2 #FAB$M_GET,T1FAB+FAB$B_FAC
F9EE' 30 05FE 254 $OPEN FAB=T1FAB,ERR=REPORT_ERROR
060F 255 BSBW ERR
0612 256
0612 257 :
0612 258 : \eventually include code here to verify file attributes.\
0612 259 :
0612 260
04 AB 00010000 8F D0 0612 261 MOVL #RAB$M_LOC,RAB$FOP(R11); locate mode
061A 262 $CONNECT RAB=T1RAB,ERR=REPORT_ERROR
F9D2' 30 062B 263 BSBW ERR
1E AB 02 90 062E 264 MOVB #RAB$C_RFA,RAB$B_RAC(R11); rfa access
59 03E8 8F 3C 0632 265 MOVZWL #1000,R9 ; starting record number
5A D4 0637 266 CLRL R10 ; as a quadword for ediv
57 D4 0639 267 CLRL R7 ; flag for 1st pass
063B 268
063B 269 :
063B 270
063B 271 T1BLOOP1:
58 08 C2 063B 272 SUBL #8,R8 ; move to previous rfa
06C5 30 063E 273 BSBW GETANDCHK
FFF2 59 FFF6 8F 0A 3D 0641 274 ACBW #10,#-10,R9,T1BLOOP1
59 0A D0 0649 275 MOVL #10,R9 ; start with record # 10
064C 276
064C 277 :
064C 278
064C 279 T1BLOOP2:
064C 280 BSBW GETANDCHK
58 08 C0 064F 281 ADDL #8,R8 ; move to next rfa
FFF2 59 0A 03DE 8F 3D 0652 282 ACBW #990,#10,R9,T1BLOOP2
065A 283
065A 284 :
065A 285
43 57 00 E2 065A 286 BBSS #0,R7,T1BDONE ; branch if 2nd pass
065E 287 $WAIT CMDORAB
066B 288 TYPE <PASS 1 O.K.>
59 03DE 8F 3C 069A 289 MOVZWL #990,R9 ; start with rec # 990
9A 11 069F 290 BRB T1BLOOP1
06A1 291 T1BDONE:
06A1 292 $WAIT CMDORAB
06AE 293 TYPE <PASS 2 O.K.>
06DD 294 $CLOSE FAB=T1FAB,ERR=REPORT_ERROR
F90F' 30 06EE 295 BSBW ERR
```



```
06F1 297
06F1 298
06F1 299      test 1c
06F1 300
06F1 301      use update to modify records created in test 1a
06F1 302
06F1 303
06F1 304      WTTYPE <START TEST 1C - $UPDATE>
FC48 CF 08 88 072D 305      BISB2 #FAB$M_UPD,T1FAB+FAB$B_FAC
      F8BA' 30 0732 306      $OPEN FAB=T1FAB,ERR=REPORT_ERROR
      02 AB B4 0743 307      BSBW ERR
      F8A5' 30 0746 308      CLRW RAB$W_ISI(R11)
      1E AB 02 90 0749 309      $CONNECT RAB=R11,ERR=REPORT_ERROR
      59 0A D0 0758 310      BSBW ERR
      58 F8DE CF DE 075B 311      MOVB #RAB$C_RFA,RAB$B_RAC(R11)
      57 D4 075F 312      MOVL #10,R9 ; start at record 10
      0762 313      MOVAL RFATBL,R8
      0767 314      CLRL R7 ; get single record
      0769 315
      0769 316 ;
      0769 317
      0769 318 T1BLOOP4:
      0769 319      BSBW GETANDCHK
      FD34 CF 059A 30 076C 320      INCL RECCNT ; modify rhb
      FD37 CF D6 0770 321      MNEGB RHBSW,RHBSW ; toggle rhb modify flag
      03 19 0777 322      BLSS 10$
      2C AB D4 0779 323      CLRL RAB$L_RHB(R11) ; don't modify rhb (default)
      55 28 AB D0 077C 324 10$: MOVL RAB$L_RBF(R11),R5 ; get record addr
      54 22 AB 3C 0780 325      MOVZWL RAB$W_RSZ(R11),R4 ; and len
      85 96 0784 326 20$: INCB (R5)+ ; modify record contents
      FB 54 F5 0786 327      SOBGTR R4,20$
      F865' 30 0789 328      $UPDATE RAB=R11,ERR=REPORT_ERROR
      2C AB FC59 CF DE 0798 329      BSBW ERR
      58 08 C0 079B 330      MOVAL RECCNT,RAB$L_RHB(R11) ; restore rhb addr
      FFBD 59 0A 03E8 8F 3D 07A1 331      ADDL #8,R8 ; bump to next rfa entry
      07A4 332      ACBW #1000,#10,R9,T1BLOOP4 ; modify every 10th record
      07AC 333
      07AC 334 ;
      07AC 335 ; now reread the modified file and check that every 10th (and only
      07AC 336 ; every 10th) record has been correctly modified
      07AC 337 ;
      07AC 338
      58 F894 CF DE 07AC 339      MOVAL RFATBL,R8
      1E AB 59 D4 07B1 340      CLRL R9 ; reset record #
      00 90 07B3 341      MOVB #RAB$C_SEQ,RAB$B_RAC(R11)
      07B7 342
      07B7 343 ;
      07B7 344      $disconnect r11
      07B7 345      bsbw err
      07B7 346      $connect r11 ; do an effective rewind
      07B7 347      bsbw err
      07B7 348
      07B7 349
      F837' 30 07B7 350      $REWIND RAB=R11,ERR=REPORT_ERROR; so do a rewind
      07C6 351      BSBW ERR
      07C9 352
      07C9 353 ;
```

```

07C9 354
07C9 355 T1BLOOP5:
57 D6 07C9 356 INCL R7 ; set switch for no rfa compare
09 DD 07CB 357 PUSHL #9 ; # gets of unmodified recs
59 D6 07CD 358 10$: INCL R9 ; bump rec #
F81F' 30 07CF 359 $GET RAB=R11,ERR=REPORT_ERROR
057A 30 07DE 360 BSBW ERR
E6 6E F5 07E1 361 BSBW CHKREC
8E D5 07E4 362 SOBGTR (SP),10$
59 D6 07E7 363 TSTL (SP)+
F803' 30 07E9 364 INCL R9 ; bump rec #
57 D4 07EB 365 $GET RAB=R11,ERR=REPORT_ERROR
FCA5 CF FCAB CF 8E 07FA 366 BSBW ERR
04 19 0806 367 CLRL R7 ; specify rfa to be checked
50 FBE7 CF 59 D6 0808 368 MNEGB RHBSW,RHBSW ; toggle rhb modified switch
03 13 0812 369 BLSS 15$
0590 30 0814 370 INCL RECCNT
0549 30 0816 371 15$: SUBL3 R9,RECCNT,R0 ; rhb contents = rec # + 1?
58 08 C0 081C 372 DECL R0
03E8 8F 59 B1 081F 373 BEQL 20$ ; branch if yes
A3 12 0824 374 BSBW BADRHB ; report error
F7C6' 30 0819 375 BSBW CHKRC1 ; check record len, rfa, and contents
08 08 081C 376 ADDL #8,R8 ; bump rfa table addr
59 B1 081F 377 CMPW R9,#1000 ; done?
A3 12 0824 378 BNEQ T1BLOOP5 ; branch if not
F7C6' 30 0826 379 $CLOSE FAB=T1FAB,ERR=REPORT_ERROR
0837 380 BSBW ERR

```

```
083A 382
083A 383
083A 384 test 2 - random i/o test for sequential file org
083A 385 test 2a
083A 386
083A 387 create a known test file of 1000 records of fixed length = 49. bytes.
083A 388 1st longword has the record # (n) followed by 45 bytes of the ascii
083A 389 character (n mod 42) + 48.
083A 390
083A 391
083A 392
083A 393
083A 394 WTTYPE <START TEST 2A - RANDOM SEQ I/O>
0876 395 T2SETUP:
0876 396 CLRW RAB$W ISI(R11)
0879 397 MOVAL CPYBUF,RAB$W_RBF(R11)
0881 398 MOVZBW #49,RAB$W_RSZ(R11)
0885 399 MOVAL T1FAB,R10
088A 400 MOVZBL #48,FAB$W_ALQ(R10)
088E 401 BICL2 #FAB$W_NAM,FAB$W_FOP(R10)
0896 402 CLRB FAB$W_FAC(R10) ; check for put default
0899 403 MOVW #FAB$C_FIX,FAB$W_RFM(R10)
089D 404 MOVW #49,FAB$W_MRS(R10) ; rec len
08A1 405 $RAB STORE RAB=(R11),ROP=<LOC,UIF>
08A9 406 $CREATE FAB=R10,ERR=REPORT_ERROR
08B8 407 BSBW ERR
08BB 408 CMPL #RMS$_SUPERSEDE,T1FAB+FAB$W_STS
08C4 409 BEQL SUPOK
08C6 410 FIELD <STATUS WORD IS NOT SUPERSEDE, THEREFORE IT>
08DB 411 SUPOK: $CONNECT RAB=R11,ERR=REPORT_ERROR
08EA 412 BSBW ERR
08ED 413
08ED 414 pre-extend file on 1st pass, put sequentially on second pass
08ED 415
08ED 416
08ED 417 BBS #FAB$W_BLK,FAB$W_RAT(R10),10$; branch if pass 2
08F2 418 MOVZWL #1000,RECCNT
08F9 419 MOVW #RAB$C_KEY,RAB$W_RAC(R11)
08FD 420 $PUT RAB=R11,ERR=REPORT_ERROR
090C 421 BSBW ERR
090F 422 BRB 20$
0911 423 10$: MOVW #RAB$C_SEQ,RAB$W_RAC(R11)
0915 424 20$: MOVZWL #1,RECCNT
091A 425
091A 426
091A 427
091A 428 NXTRC2: EDIV #42,RECCNT,R2,R7 ; compute char for record
0921 429 ADDB2 #48,R7 ; make it ascii
0924 430 MOVL RECCNT,CPYBUF ; insert rec #
092D 431 MOVCS #0,(SP),R7,#45,CPYBUF+4 ; fill rec with char
0932 432
0937 433 $PUT RAB=R11,ERR=REPORT_ERROR
0946 434 BSBW ERR
0949 435 BSBW CHKRFA
094C 436
094C 437 : print message every 100 records

28 AB 00000000'EF B4 0876 395
22 AB 31 9B 0879 396
5A FADB CF DE 0881 397
10 AA 30 9A 0885 398
04 AA 01000000 8F CA 088A 399
16 AA 94 088E 400
1F AA 01 90 0896 401
36 AA 31 B0 0899 402
F745' 30 089D 403
FAAB CF 00010631 8F D1 08A1 404
15 13 08A9 405
F713' 30 08B8 406
08BB 407
08C4 408
08C6 409
08DB 410
08EA 411
08ED 412
08ED 413
08ED 414
08ED 415
08ED 416
1F 1E AA 03 E0 08ED 417
FAFF CF 03E8 8F 3C 08F2 418
1E AB 01 90 08F9 419
F6F1' 30 08FD 420
04 11 090C 421
1E AB 00 90 090F 422
FADE CF 01 3C 0911 423
0915 424
091A 425
091A 426
091A 427
57 52 FAD9 CF 2A 7B 091A 428
57 30 80 0921 429
00000000'EF FAD0 CF D0 0924 430
2D 57 6E 00 2C 092D 431
00000004'EF 0932 432
F6B7' 30 0937 433
05A1 30 0946 434
0949 435
094C 436
094C 437
```


52	FAA3 CF	00000064	8F	7B	094C 438 ;		
			57		094C 439		
			57		094C 440	EDIV	#100,RECCNT,R2,R7
			03	D5	0956	TSTL	R7
			0537	12	0957 441	BNEQ	T2CNT
				30	0959 442	BSBW	TYPRFA
					095B 443		
					095E 444		
					095E 445 ;		
FFB2	FA92 CF	01	03E8	8F	095E 446	T2CNT:	ACBW #1000,#1,RECCNT,NXTRC2
			F686'	3D	095E 447		\$CLOSE FAB=R10,ERR=REPORT_ERROR
			02 AB	B4	0968 448		BSBW ERR
					0977 449		CLRW RAB\$W,ISI(R11)
					097A 450		WTTYPE <END STEP 2A - START STEP 2B>
					097D 451		

```
09B9 453
09B9 454
09B9 455 : test 2b
09B9 456 :
09B9 457 : reread file created in step 2a and try random and sequential
09B9 458 : access via $get
09B9 459 :
09B9 460
04 AA 16 AA 09 90 09B9 461 MOVB #FAB$M_PUT!FAB$M_UPD,FAB$B_FAC(R10); upd implies get access
01000000 8F C8 09BD 462 BISL2 #FAB$M_NAM,FAB$B_FOP(R10)
F629' 30 09C5 463 $OPEN FAB=R10,ERR=REPORT_ERROR
09D4 464 BSBW ERR
09D7 465
09D7 466 :
09D7 467 : movl #rab$m_loc,rab$l_rop(r11)
09D7 468 :
09D7 469 :
09D7 470 $RAB_STORE RAB=(R11),ROP=<LOC,UIF>
09DF 471 MOVB #RAB$C_KEY,RAB$B_RAC(R11)
09E3 472 $CONNECT RAB=R11,ERR=REPORT_ERROR
F60B' 30 09F2 473 BSBW ERR
F9FC CF 03E8 8F 3C 09F5 474 MOVZWL #1000,RECCNT
09FC 475
09FC 476 :
09FC 477 : get all records in reverse order
09FC 478 :
09FC 479
FFF3 F9F1 CF FFFF 8F 054D 30 09FC 480 10$: BSBW GTCHK2
02 3D 09FF 481 ACBW #2,#-1,RECCNT,10$
0A09 482
0A09 483 :
0A09 484 : now get them all forward
0A09 485 :
0A09 486 :
FFF3 F9E4 CF 01 03E8 8F 0540 30 0A09 487 20$: BSBW GTCHK2
03E8 8F 3D 0A0C 488 ACBW #1000,#1,RECCNT,20$
0A16 489
0A16 490 :
0A16 491 : now get every 10th record in reverse order followed by the
0A16 492 : next 10 in sequential order
0A16 493 :
0A16 494 :
F9DB CF 03DE 8F 80 0A16 495 MOVW #990,RECCNT
052C 30 0A1D 496 30$: BSBW GTCHK2
1E AB 00 90 0A20 497 MOVB #RAB$C_SEQ,RAB$B_RAC(R11)
58 0A 00 0A24 498 MOVL #10,R8
F9CD CF D6 0A27 499 25$: INCL RECCNT
051E 30 0A2B 500 BSBW GTCHK2
F6 58 F5 0A2E 501 SOBGTR R8,25$
1E AB 01 90 0A31 502 MOVB #RAB$C_KEY,RAB$B_RAC(R11)
FFDE F9BB CF FFEC 8F 01 3D 0A35 503 ACBW #1,#-20,RECCNT,30$
0A3F 504
0A3F 505 :
0A3F 506
0A3F 507 WTTYPE <PASS 1 O.K.>
```

```
0A7B 509
0A7B 510
0A7B 511 : do 10 random puts, changing record contents
0A7B 512 :
0A7B 513 :
0A7B 514
0A7E 515      MOVB    #^A/O/,R6                ; updating character
0A84 516      MOVZBL  #91,RECCNT
40$: 517      MOVL    RECCNT,CPYBUF
      MOVCS    #0,(SP),R6,#45,CPYBUF+4 ; change the record
0A92 518
0A97 519      $PUT    RAB=R11,ERR=REPORT_ERROR
0AA6 520      BSBW    ERR
0AA9 521      BSBW    CHKRFA
0AAC 521      ACBW    #1000,#101,RECCNT,40$
0AB6 522
0AB8 523      BSBW    CHKMOD                ; go verify changes
0ABB 524
0ABB 525 : do 10 random get/update pairs
0ABB 526 :
0ABB 527 :
0ABB 528
0ABE 529      MOVB    #^A/I/,R6                ; updating character
0AC4 530      MOVZBL  #91,RECCNT                ; starting rec #
50$: 531      $GET    RAB=R11,ERR=REPORT_ERROR
      BSBW    ERR
0AD3 532      MOVAL   CPYBUF+4,R8                ; get addr of record char
0AD6 533      MOVL    #45,R9
0ADD 534      55$:   INCB    (R8)+                ; bump contents
0AE0 535      SOBGTR  R9,55$
0AE2 536      $UPDATE RAB=R11,ERR=REPORT_ERROR
0AE5 537      BSBW    ERR
0AF4 538      BSBW    CHKRFA
0AF7 539      ACBW    #1000,#101,RECCNT,50$
0AFA 540
0B04 541      BSBW    CHKMOD                ; go verify changes
0B06 542 :
0B09 543 :
0B09 544
0B09 545      WTTYPE   <PASS 2 O.K.>
0B45 546      $CLOSE  FAB=R10,ERR=REPORT_ERROR
0B54 547      BSBW    ERR
3F 1E AA 03 E2 0B57 548      BBSS    #FAB$V_BLK,FAB$B_RAT(R10),DONE
0B5C 549      WTTYPE   <DUPLICATE TEST WITH RECORDS NOT CROSSING BLOCK BOUNDARIES>
FCDB 31 0B98 550      BRW     T2SETUP
0B9B 551 :
0B9B 552 :
0B9B 553      DONE:   MOVB    #FAB$M_PUT,FAB$B_FAC(R10)
0B9F 554      MOVB    #RAB$C_SEQ,RAB$B_RAC(R11)
0BA3 555      $WAIT   CMDORAB
0BB0 556      MOVL    #RAB$M_WBH,RAB$L_ROP+CMDORAB
0BBB 557      BICL2    #FAB$M_NAM,T1FAB$FAB$L_FOP
0BC4 558
0BC4 559
0BC4 560 :
0BC4 561 :BUT FIRST -- TEST 3
0BC4 562 :
```


			OBC4	563	:	test truncate
			OBC4	564	:	
			OBC4	565	:	
			OBC4	566	:	\$FAB_STORE FAB=R10,SHR=#0,FAC=<PUT,GET,TRN>; set up for trn
			OBCB	567	:	\$OPEN FAB=R10,ERR=REPORT_ERROR
F423'	30		OBDA	568	:	BSBW ERR
			OBDD	569	:	\$CONNECT RAB=R11,ERR=REPORT_ERROR
F411'	30		OBEC	570	:	BSBW ERR
			OBFE	571	:	\$GET RAB=R11,ERR=REPORT_ERROR; get 1st record, setting up cp
F3FF'	30		OBFE	572	:	BSBW ERR
			OC01	573	:	\$TRUNCATE RAB=R11,ERR=REPORT_ERROR; truncate all
F3ED'	30		OC10	574	:	BSBW ERR
			OC13	575	:	\$GET RAB=R11 ; should get eof
0001827A 8F	50	D1	OC1C	576	:	CMPL R0,#RMS\$_EOF
	06	13	OC23	577	:	BEQL TROK
5A	5B	D0	OC25	578	:	MOVL R11,R10
F3D5'	30		OC28	579	:	BSBW EOFPUT ; too bad
			OC2B	580	:	TYPE <TRUNCATE SUCCEEDED!>
			OC5A	581	:	\$DISCONNECT RAB=R11,ERR=REPORT_ERROR; clean up
F394'	30		OC69	582	:	BSBW ERR
			OC6C	583	:	\$CLOSE FAB=R10,ERR=REPORT_ERROR
F382'	30		OC7B	584	:	BSBW ERR
			OC7E	585	:	\$FAB_STORE FAB=R10,SHR=<PUT,GET,UPI>,FAC=PUT
1F AA	03	90	OC87	586	:	MOVB #FAB\$_VFC,FAB\$_RFM(R10)
36 AA	64 8F	9B	OC8B	587	:	MOVZBW #100,FAB\$_MRS(R10)
1E AA	02	90	OC90	588	:	MOVB #FAB\$_CR,FAB\$_RAT(R10)
10 AA	30	D0	OC94	589	:	MOVL #48,FAB\$_ALQ(R10)
3F AA	04	90	OC98	590	:	MOVB #4,FAB\$_FSZ(R10)
28 AB	00000000'EF	DE	OC9C	591	:	MOVAL CPYBUF,RAB\$_RBF(R11)
			OCA4	592	:	
			OCA4	593	:	
			OCA4	594	:	try to erase the file!
			OCA4	595	:	
			OCA4	596	:	
			OCA4	597	:	\$ERASE FAB=T1FAB,ERR=REPORT_ERROR
F348'	30		OCB5	598	:	BSBW ERR
			OCB8	599	:	TYPE <T1FILE.DAT HAS BEEN ERASED>
F678 CF	00100000 8F	C8	OCF7	600	:	BISL2 #FAB\$_CTG,FAB\$_FOP+T1FAB; for 'f' test
			OCF0	601	:	FINISH <SEQUENTIAL TESTS>
		04	OD05	602	:	RET

```

      0D06 604
      0D06 605
      0D06 606 : subroutine to read in a record
      0D06 607 :
      0D06 608
      0D06 609 GETANDCHK:
10 AB 68 06 28 0D06 610 MOV C3 #6,(R8),RAB$W_RFA(R11) ; rfa to rab
      15 57 E8 0D0B 611 BLBS R7,PASS2 ; branch if pass 2
      F2E0' 30 0D0E 612 $GET RAB=R11,ERR=REPORT_ERROR; get via rfa
      3C 10 0D1D 613 BSBW ERR
      05 0D20 614 BSBB CHKREC
      0D22 615 RSB
      0D23 616 PASS2:
      1E AB 02 90 0D23 617 MOV B #RAB$C_RFA,RAB$B_RAC(R11)
      F2C7' 30 0D27 618 $FIND RAB=R1T,ERR=REPORT_ERROR
      1E AB 00 90 0D36 619 BSBW ERR
      0A DD 0D39 620 MOV B #RAB$C_SEQ,RAB$B_RAC(R11); switch back to sequential
      F2AF' 30 0D3D 621 PUSHL #10 ; loop count
      0B 10 0D3F 622 T1BLOOP3:
      59 D6 0D4E 623 $GET RAB=R11,ERR=REPORT_ERROR
      E7 6E F5 0D51 624 BSBW ERR
      59 0A C2 0D53 625 BSBB CHKREC
      8E D5 0D55 626 INCL R9 ; bump record count
      05 0D58 627 SORGTR (SP),T1BLOOP3
      0D5B 628 SUBL2 #10,R9 ; restore record count
      0D5D 629 TSTL (SP)+ ; clean up stack
      0D5D 630 RSB
```

```

      56  52  59  00000064 8F 7B 0D63 639  CHKREC:  CMPL  R9,RECCNT          ; fixed header = record #?
      55  52  F67F CF 0A 7B 0D74 643  BNEQ  BADRHB          ; compute record length
      6E  00  55  28 8B 56 2D 0D7E 645  CMPW  R6,RAB$W_RSZ(R11) ; = rms record len?
      10 AB  68 07 57 E8 0D87 647  BNEQ  BADRSZ          ; compute character
      01 05 0D91 650 10$:  ADDB  #48,R5          ; make ascii
      06 01 0D8F 649  CMPC5 R6,RAB$L_RBF(R11),R5,#0,(SP); match the record?
      07 06 0D8A 648  BNEQ  BADRBF          ; branch if pass 2
      08 07 0D87 647  BLBS  R7,10$          ; rfa the same?
      09 08 0D84 646  CMPC3 #6,(R8),RAB$W_RFA(R11)
      10 09 0D81 645  BNEQ  BADRFA
      11 10 0D7E 644  RSB

```



```

OD92 652
OD92 653
OD92 654 :: handle errors
OD92 655 ::
OD92 656
OD92 657 BADRFA: FIELD <RFA>
46 11 ODA7 658 BRB ERROR
OD92 659 BADRHB: FIELD <RHB>
2F 11 ODBE 660 BRB ERROR
OD92 661 BADRSZ: FIELD <RSZ>
18 11 ODD5 662 BRB ERROR
OD92 663 BADRBF: FIELD <RBF>
01 11 ODEC 664 BRB ERROR
OD92 665
OD92 666 :
OD92 667
05 ODEE 668 DONT: RSB
ODEF 669 ERROR:
F8 00000000'EF E9 ODEF 670 BLBC VERBOSITY,DONT
00000004'EF 01 CA ODF6 671 $WAIT RAB=CMDORAB
ODE0 672 BICL2 #RAB$M_ASY,RAB$L_RDP+CMDORAB
ODE0 673 $FAO S T2STR,CMDORAB+RAB$W_RSZ,FAOBUF,-
ODE0 674 R9,RECCNT,R6,RAB$W_RSZ(R11),-
ODE0 675 (R8),4(R8),RAB$W_RFA(R11),RAB$W_RFA+4(R11)
00000028'EF FF 50 E9 OE37 676 BLBC RO,-
00000000'EF 9E OE3A 677 MOVAB CMDBUF,CMDORAB+RAB$L_RBF
F1A5' 30 OE45 678 $PUT RAB=CMDORAB,ERR=REPORT_ERROR
50 22 AB 3C OE58 679 BSBW ERR
OE5B 680 MOVZWL RAB$W_RSZ(R11),RO
OE5F 681 $FAO S T3STR,CMDORAB+RAB$W_RSZ,FAOBUF,-
OE5F 682 RO,RAB$L_RBF(R11)
FF 50 E9 OE7B 683 BLBC RO,-
OE7E 684 $PUT RAB=CMDORAB,ERR=REPORT_ERROR
F16C' 30 OE91 685 BSBW ERR
05 OE94 686 RSB
OE95 687
OE95 688 ::
OE95 689 :: output a message
OE95 690 ::
OE95 691
50 00000000'EF E9 OE95 692 TYPRFA: BLBC VERBOSITY,NO
OE9C 693 $WAIT RAB=UTRAB
OE9C 694 $FAO S T1STR,CMDORAB+RAB$W_RSZ,FAOBUF,RECCNT,RAB$W_RFA(R11),-
OE9C 695 RAB$W_RFA+4(R11)
00000028'EF FF 50 E9 OEC8 696 BLBC RO,-
00000000'EF 9E OECB 697 MOVAB CMDBUF,CMDORAB+RAB$L_RBF
F114' 30 OED6 698 $PUT RAB=CMDORAB,ERR=REPORT_ERROR
05 OEE9 699 BSBW ERR
OEEC 700 NO: RSB
```

```

52 51 50 49 48 47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1 0
OEED 702
OEED 703
OEED 704
OEED 705
OEED 706
OEED 707
OEED 708
50 F506 CF 01 C3 OEED 709 CHKRFA: SUBL3 #1,RECCNT,R0 ; get rec # - 1
10 1E AA 03 E0 OEF3 710 BBS #FAB$V_BLK,FAB$B_RAT(R10),10$
51 00 32 50 7A OEF8 711 EMUL R0,#50,#0,R1 ; get byte addr
51 00000200 8F 7B OEF9 712 EDIV #512,R1,R1,R2 ; compute vbn-1 & offset
OA 11 OF06 713 BRB 20$
OF08 714
OF08 715
OF08 716
OF08 717
OF08 718
OF08 719 10$: CLRL R1 ; zero extend record #
52 51 50 0A 7B OF0A 720 EDIV #10,R0,R1,R2 ; compute vbn-1 & rec # in blk
52 32 A4 OF0F 721 MULW2 #50,R2 ; compute offset
51 D6 OF12 722 20$: INCL R1 ; vbn
10 AB 51 D1 OF14 723 CMPL R1,RAB$W_RFA(R11)
07 12 OF18 724 BNEQ ERRRFA
14 AB 52 B1 OF1A 725 CMPW R2,RAB$W_RFA+4(R11)
01 12 OF1E 726 BNEQ ERRRFA
05 OF20 727 RSB
OF21 728 ERRRFA: WFIELD <RFA>
OF43 729 MBPT
05 OF4B 730 RSB

```

```

OF4C 732
OF4C 733
OF4C 734
OF4C 735
OF4C 736
57 50 F4A7 CF 2A 7B OF4C 737 GTCHK2: EDIV #42,RECCNT,R0,R7 ; compute char
      57 30 80 OF53 738 ADDB #48,R7
      F098' 30 OF56 739 GTCHK2A: $GET RAB=R11,ERR=REPORT_ERROR
      FF82 30 OF65 740 BSBW ERR
      52 28 AB D0 OF68 741 BSBW CHKRFA
      F484 CF 82 D1 OF6B 742 MOVL RAB$RBF(R11),R2 ; get rec addr
      09 12 OF6F 743 CMPL (R2)+,RECCNT ; recnt o.k.?
6E 00 57 62 2D 12 OF74 744 BNEQ ERRREC
      01 12 OF76 745 CMPC5 #45,(R2),R7,#0,(SP) ; match?
      05 05 OF7C 746 BNEQ ERRREC
      OF7E 747 RSB
      OF7F 748 ERRREC: WFIELD <RECORD CONTENTS>
      OFA1 749 MBPT
      OFA9 750 RSB
      OFAA 751
      OFAA 752
      OFAA 753
      OFAA 754 ; subroutine to verify that updated records were changed but not
      OFAA 755 ; the preceeding or following records
      OFAA 756
      OFAA 757
      OFAA 758
      OFAA 759 CHKMOD: MOVZBL #90,RECCNT ; starting rec #
      F448 CF 5A 8F 9A OFAA 760 CHKNXT: BSBW GTCHK2
      FF99 30 OFB0 761 INCL RECCNT
      F441 CF D6 OFB3 762 MOVL R6,R7 ; check character
      57 56 D0 OFB7 763 BSBW GTCHK2A
      FF99 30 OFBA 764 INCL RECCNT
      F437 CF D6 OFBD 765 CMPW RECCNT,#1001 ; all done?
      F433 CF B1 OFC1 766 BEQL 10$
      03E9 8F 0C 13 OFC8 767 BSBW GTCHK2
      FF7F 30 OFCA 768 ADDW #99,RECCNT
      F424 CF 0063 8F A0 OFCD 769 BRB CHKNXT
      DA 11 OFD4 770 10$: $GET R11
      0001827A 8F 50 D1 OFDF 771 CMPL R0,#RMS$_EOF
      01 12 OFE6 772 BNEQ 20$
      05 05 OFE8 773 RSB
      OFE9 774 20$: $WAIT RAB=CMDORAB
      OFE9 775 PUSHL R10 ; save it
      5A 5A DD OFF6 776 MOVL R11,R10 ; bad structure!!!
      5A 5B D0 OFF8 777 BSBW EOFPUT
      F002' 30 OFFB 778 POPR #^M<R10>
      0400 8F BA OFFE 779 MBPT
      1002 780 RSB
      100A 781
      100B 782
      100B 783 .END
```


RMSTEST1
Symbol table

GENERAL RMS TEST PROGRAM ;

F 6

16-SEP-1984 01:45:37 VAX/VMS Macro V04-00
5-SEP-1984 04:21:39 [UETP.SRC]RMSTEST1.MAR;1

Page 19
(24)

```
== 00000000
== 000003B4 R D 01
== 000003F8 R D 01
== 00000043 D
== 00000001 D
== 000000EF D
== 00000187 R D 04
== 0000000F D
== 0000001E D
== 00000010 D
== 00000008 D
== 00000004 D
== 00000006 D
== 00000000 D
== 00000002 D
== 00000001 D
== 00000003 D
== 00000003 D
== 00000001 D
00000DD7 R D 01
00000D92 R R D 01
00000DA9 R R D 01
00000DC0 R D 01
***** X 01
***** X 01
00000FAA R R D 01
00000FB0 R R D 01
00000D65 R R D 01
00000D5E R R D 01
00000EED R D 01
***** X 01
***** X 01
***** X 01
00000B9B R D 01
00000DEE R D 01
***** X 01
***** X 01
00000DEF R R D 01
00000F7F R R D 01
00000F21 R D 01
== 00000016 D
== 00000034 D
== 0000003F D
== 0000001E D
== 0000001F D
== 00000017 D
== 00000003 D
== 00000050 D
== 00000001 D
== 00000000 D
== 00000003 D
== 00000010 D
== 0000002C D
== 00000004 D
== 00000008 D
== 00000002 D
```

```
FABSM_CTG
FABSM_GET
FABSM_NAM
FABSM_PUT
FABSM_UPD
FABSV_BLK
FABSV_CHAN_MODE
FABSV_CR
FABSV_CTG
FABSV_FILE_MODE
FABSV_GET
FABSV_LNM_MODE
FABSV_PUT
FABSV_SUP
FABSV_TRN
FABSV_UPI
FABSW_GBC
FABSW_MRS
FAOBUF
FINPUT
FIN_DESCR
FLDPUT
FLD_DESCR
GETANDCHK
GTCHK2
GTCHK2A
NAMBLK
NO
NXTRC2
NXTREC
PASS2
RABSB_RAC
RABSC_BID
RABSC_BLN
RABSC_KEY
RABSC_RFA
RABSC_SEQ
RABSL_CTX
RABSL_RBF
RABSL_RHB
RABSL_ROP
RABSM_ASY
RABSM_LOC
RABSM_WBH
RABSV_LOC
RABSV_UIF
RABSV_WBH
RABSW_ISI
RABSW_RFA
RABSW_RSZ
RECCNT
REPORT_ERROR
RFATBL
RHBSW
RMSS_EOF
RMSS_SUPERSEDE
RMTSTEST_1A
```

```
= 00100000 D
= 00000002 D
= 01000000 D
= 00000001 D
= 00000008 D
= 00000003 D
= 00000002 D
= 00000001 D
= 00000014 D
= 00000004 D
= 00000001 D
= 00000000 D
= 00000002 D
= 00000004 D
= 00000006 D
= 00000048 D
= 00000036 D
***** X 01
***** X 01
***** X 01
***** X 01
00000D06 R D 01
00000F4C R R D 01
00000F56 R R D 01
***** X 01
00000EEC R R D 01
0000091A R R D 01
00000511 R R D 01
00000D23 R D 01
= 0000001E D
= 00000001 D
= 00000044 D
= 00000001 D
= 00000002 D
= 00000000 D
= 00000018 D
= 00000028 D
= 0000002C D
= 00000004 D
= 00000001 D
= 00010000 D
= 00000400 D
= 00000010 D
= 00000004 D
= 0000000A D
= 00000002 D
= 00000010 D
= 00000022 D
000003F8 RG D 01
***** X 01
00000044 R D 01
000004AB R D 01
= 0001827A D
= 00010631 D
000004AC RG D 01
```

RMSTEST1
Symbol table

GENERAL RMS TEST PROGRAM ;

G 6

16-SEP-1984 01:45:37 VAX/VMS Macro V04-00
5-SEP-1984 04:21:39 [UETP.SRC]RMSTEST1.MAR;1

Page 20
(24)

SUPOK	000008DB	R	D	01
SYSS\$CLOSE	*****	GX		01
SYSS\$CONNECT	*****	GX		01
SYSS\$CREATE	*****	GX		01
SYSS\$DISCONNECT	*****	GX		01
SYSS\$ERASE	*****	GX		01
SYSS\$FAO	*****	X		01
SYSS\$FIND	*****	GX		01
SYSS\$GET	*****	GX		01
SYSS\$OPEN	*****	GX		01
SYSS\$PUT	*****	GX		01
SYSS\$REWIND	*****	GX		01
SYSS\$TRUNCATE	*****	GX		01
SYSS\$UPDATE	*****	GX		01
SYSS\$WAIT	*****	GX		01
T1BDONE	000006A1	R	D	01
T1BLOOP1	0000063B	R	D	01
T1BLOOP2	0000064C	R	D	01
T1BLOOP3	00000D3F	R	D	01
T1BLOOP4	00000769	R	D	01
T1BLOOP5	000007C9	R	D	01
T1CNT	00000556	R	D	01
T1FAB	00000364	RG	D	01
T1L	= 00000025		D	
T1RAB	00000384	RG	D	01
T1S	00000408	R	D	01
T1START	00000000	RG	D	01
T1STR	00000400	R	D	01
T2CNT	0000095E	R	D	01
T2L	= 0000005F		D	
T2S	00000435	R	D	01
T2SETUP	00000876	R	D	01
T2STR	0000042D	R	D	01
T3L	= 0000000F		D	
T3S	0000049C	R	D	01
T3STR	00000494	R	D	01
TROK	00000C2B	R	D	01
TYPRFA	00000E95	R	D	01
VERBOSITY	*****	X		01
WTRAB	00000000	RG	D	01

! Psect synopsis !

PSECT name	Allocation	PSECT No.	Attributes
ABS	00000000 (0.)	00 (0.)	NOPIC USR CON ABS LCL NOSHR NOEXE NORD NOWRT NOVEC BYTE
RMSTEST	0000100B (4107.)	01 (1.)	NOPIC USR CON REL GBL NOSHR EXE RD WRT NOVEC LONG
\$ABSS	00000000 (0.)	02 (2.)	NOPIC USR CON ABS LCL NOSHR EXE RD WRT NOVEC BYTE
\$RMSNAM	00000015 (21.)	03 (3.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE
__RMSNAM	00000196 (406.)	04 (4.)	NOPIC USR CON REL LCL NOSHR EXE RD WRT NOVEC BYTE

! Performance indicators !

Phase	Page faults	CPU Time	Elapsed Time
-----	-----	-----	-----
Initialization	38	00:00:00.07	00:00:00.55
Command processing	133	00:00:00.54	00:00:02.18
Pass 1	376	00:00:17.01	00:00:35.60
Symbol table sort	0	00:00:00.74	00:00:01.42
Pass 2	138	00:00:03.96	00:00:06.21
Symbol table output	18	00:00:00.13	00:00:00.17
Psect synopsis output	2	00:00:00.03	00:00:00.12
Cross-reference output	0	00:00:00.00	00:00:00.00
Assembler run totals	707	00:00:22.49	00:00:46.25

The working set limit was 1650 pages.
86298 bytes (169 pages) of virtual memory were used to buffer the intermediate code.
There were 30 pages of symbol table space allocated to hold 618 non-local and 34 local symbols.
783 source lines were read in Pass 1, producing 58 object records in Pass 2.
69 pages of virtual memory were used to define 50 macros.

! Macro library statistics !

Macro library name	Macros defined
-----	-----
_\$255\$DUA28:[SYS.OBJ]LIB.MLB;1	0
_\$255\$DUA28:[SYSLIB]STARLET.MLB;2	36
TOTALS (all libraries)	36

1074 GETS were required to define 36 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS\$:RMSTEST1/OBJ=OBJ\$:RMSTEST1 MSRC\$:RMSTEST1/UPDATE=(ENH\$:RMSTEST1)+EXECML\$/LIB

0409

**DIGITAL
CONFIDE**